AMIGO: ADAPTIVE MULTILINGUAL INSTRUCTION TUNING VIA GRADIENT SIMILARITY AND BUDGETED OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multilingual instruction tuning requires allocating a limited training budget across languages whose cross-lingual interactions evolve throughout training. Prior work typically addresses data mixing (e.g., sampling ratios) and training strategy (e.g., curricula or scheduling) separately, often relying on static or heuristic decisions that do not account for the non-stationary nature of multilingual optimization. We propose **AMIGO** (Adaptive Multilingual Instruction Tuning via Gradient Similarity and Budgeted Optimization), a unified framework that couples language grouping, training order, and budget allocation through gradient geometry re-estimated as training progresses. AMIGO operates in regimes: at each stage, it clusters languages based on validation gradient similarity, selects the next language family via a compatibility score that balances optimization potential with transition smoothness, and allocates training budget using a continuous knapsack objective grounded in a first-order approximation of the multilingual loss. The utility of a language is defined by its need-weighted gradient alignment with out-of-family languages, while its cost is derived from validation loss and gradient norms, eliminating the need for external evaluation at regime boundaries. This formulation captures the dynamic nature of cross-lingual interactions and enables adaptive reallocation of training resources as optimization progresses. Empirical evaluations on various instruction-following sets across 23 languages and multiple model families demonstrate that AMIGO outperforms existing data-mixing and data-selection baselines on average under fixed training budgets, with notable gains on several benchmarks.

1 INTRODUCTION

Training large language models (LLMs) via multilingual instruction tuning requires allocating a finite training budget across languages whose cross-lingual interactions evolve throughout optimization. Although joint training facilitates cross-lingual transfer, it can also introduce interference, making the question of *which languages to train together and how much budget to allocate to each* central to achieving strong multilingual performance. Prior work has largely addressed these two questions separately: data-mixing strategies fix sampling ratios before training (Guo et al., 2025; Chen et al., 2023), while curriculum methods determine training order through fixed schedules (Zhang et al., 2021; Yoo et al., 2025). Both families of approaches share a common assumption, that the relative utility of each language remains approximately constant throughout training, which we argue is fundamentally at odds with the optimization dynamics of multilingual instruction tuning.

The interactions between languages during joint training are governed by the alignment of their loss gradients (Wang et al., 2023). A key observation, formalized as Theorem K.1 in the Appendix, is that gradient similarity is not a fixed property of a language pair but a function of the current model parameters: any parameter update, regardless of which language drives it, rotates the gradient directions of all other languages via Hessian–gradient interactions. As a result, language groupings that were beneficial early in training may become suboptimal or even harmful later, and the optimal sampling ratios shift accordingly. Recent methods such as PiKE (Li et al., 2025), GREATS (Wang et al., 2024a), and ADAPT (Kadasi et al., 2025) have empirically observed that sampling ratios change substantially during training, but do not identify gradient non-stationarity as the underlying cause or respond to it at the level of language grouping and training order.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This non-stationarity has implications for all three dimensions of multilingual training design. First, language groupings must be re-estimated periodically rather than fixed upfront, since the gradient-compatibility structure that determines which languages benefit from joint training evolves as parameters update (see Figure 1, discussed in Section 5.2). Second, the order in which language families are trained matters: transitioning abruptly to a family whose gradient direction conflicts with the current parameter state can destabilize optimization, while a smooth curriculum through compatible families enables steady progress. Third, within each training stage, budget allocation across languages must reflect their current utility and learning speed rather than static proxies such as performance on held-out benchmarks, or pre-computed scaling laws (Guo et al., 2025; Chen et al., 2023), which introduce overhead and lag behind the true optimization state.

We introduce AMIGO (Adaptive Multilingual Instruction Tuning via Gradient Similarity and Budgeted Optimization), a unified framework that addresses all three dimensions jointly. AMIGO operates in discrete *regimes*: at each regime boundary, it re-clusters languages based on their current validation gradients, selects the next language family via a compatibility score that balances optimization potential with transition smoothness, and allocates the regime budget within the active family using a continuous knapsack objective grounded in a first-order approximation of the multilingual loss. The utility of training a language is measured by its need-weighted gradient inner product with out-of-family languages, which captures cross-lingual transfer potential, while its cost is derived from validation loss and gradient norms, reflecting current learning difficulty without requiring external evaluation. Crucially, these three components are not independent modules but jointly informed by the same gradient information computed at each regime boundary, making the framework both principled and computationally practical.

To summarize, we make the following contributions:

- We introduce a regime-based framework for multilingual instruction tuning that adapts training decisions to the evolving cross-lingual gradient structure, enabling dynamic grouping, ordering, and budget allocation across languages.
- We propose a gradient-based language grouping mechanism that clusters languages according to validation gradient similarity, and empirically show that these groupings are inherently non-stationary during training, motivating periodic re-estimation.
- We design a compatibility-driven regime selection strategy that chooses the next language family by balancing optimization potential and transition smoothness, ensuring stable progression across regimes.
- Within each regime, we formulate budget allocation as a continuous knapsack problem with diminishing returns, where language utility is defined via need-weighted gradient alignment and cost is derived from validation loss and gradient norms.
- We conduct large-scale empirical evaluations under fixed training budgets, demonstrating that AMIGO outperforms existing data-mixing and data-selection baselines on average, with consistent gains across language families.

2 Related Work

Data Allocation. Prior work has explored data allocation at multiple granularities, ranging from domain-level reweighting guided by validation loss (Fan et al., 2024; Kang et al., 2025) to token-level selection strategies that prioritize high-utility training signals (Lin et al., 2024). Complementary approaches rely on heuristic filtering to remove noisy data (Raffel et al., 2020; Laurençon et al., 2022), as well as deduplication and controlled duplication to improve efficiency and generalization (Lee et al., 2022; Abbas et al., 2023). In multilingual settings, recent studies explicitly model cross-lingual transfer to guide language-wise allocation (Guo et al., 2025). Beyond static schemes, adaptive batching and data mixing methods dynamically adjust training distributions based on optimization dynamics (Li et al., 2025; Wang et al., 2024a), while Pareto-front–based formulations have been explored for balancing trade-offs in multilingual neural machine translation (Chen et al., 2023). DynamixSFT (Shin et al., 2026) treats each instruction-tuning dataset as a bandit arm and adapts dataset sampling probabilities with a one-step look-ahead reward. AMIGO works at the level of languages and also regroups them as training proceeds.

Multi-task Optimization. Multi-task and multilingual optimization has extensively studied the role of gradient interactions during training. Several works leverage gradient alignment to guide adaptive data mixing or batch selection (Wang et al., 2024a; Li et al., 2025). A related line of research focuses on mitigating negative transfer by resolving gradient conflicts, including methods such as PCGrad (Yu et al., 2020) and Gradient Vaccine (Wang et al., 2021), which show that gradient similarity correlates with task-relatedness and performance. Building on these insights, Wang et al. (2023) proposes grouping languages based on gradient similarity for joint training. TAG (Fifty et al., 2021) measures how a gradient step on one task changes another task's loss and selects task groups from a single joint training run. AMIGO re-estimates the grouping at every regime boundary, chooses which group to train next, and allocates the budget among the languages in that group.

3 MOTIVATION

Gradient Alignment Governs Cross-Lingual Transfer. Let $\theta \in \mathbb{R}^d$ denote the shared model parameters, and let each language ℓ induce a loss function $L_\ell(\theta)$. Multilingual instruction tuning optimizes a weighted objective $\mathcal{L}(\theta) = \sum_{\ell=1}^{L} w_\ell L_\ell(\theta)$, where $w_\ell \geq 0$ and $\sum_\ell w_\ell = 1$. A single gradient descent step with learning rate η updates the parameters as $\theta^+ = \theta - \eta \sum_{\ell=1}^{L} w_\ell \nabla L_\ell(\theta)$. Using a first-order Taylor approximation, the change in loss for language i satisfies

$$\Delta L_i \approx -\eta \sum_{\ell=1}^{L} w_\ell \langle \nabla L_i(\theta), \nabla L_\ell(\theta) \rangle. \tag{1}$$

This decomposes into a self-improvement term $-\eta w_i \|\nabla L_i(\theta)\|^2$ and cross-lingual interaction terms. Positive inner products correspond to cooperative updates that reduce L_i; negative inner products indicate interference. Loss gradient alignment therefore provides a principled, optimization-space measure of cross-lingual compatibility, one that does not require linguistic priors or typological similarity.

Gradient Similarity Is Inherently Non-Stationary. Crucially, gradient alignment is not a fixed property of a language pair but a function of the current parameters. Since all languages share θ, any update regardless of which language drives it, alters the gradient landscape for all others. After an update driven by language j, $\theta^+ = \theta - \eta \nabla L_j(\theta)$, a first-order Taylor expansion yields

$$\nabla L_i(\theta^+) \;\approx\; \nabla L_i(\theta) \;-\; \eta\, \nabla^2 L_i(\theta)\, \nabla L_j(\theta), \tag{2}$$

where $\nabla^2 L_i(\theta)$ is the Hessian of L_i. The gradient direction of language i rotates under every parameter update via Hessian–gradient interactions, causing inner products $\langle \nabla L_i(\theta), \nabla L_j(\theta) \rangle$ and hence cosine-based similarities, to evolve continuously. This effect compounds across training: languages that initially interfere may become aligned, and initially synergistic pairs may begin to conflict. We formalize this in Theorem K.1 (Appendix), which establishes that gradient geometry necessarily changes across regimes whenever Hessian–gradient interactions are non-trivial.

Implications for Multilingual Training Design. The non-stationarity of gradient similarity has consequences across all three dimensions of multilingual training:

1. *Language grouping.* Methods that fix language clusters upfront, based on linguistic similarity or a one-time gradient measurement, operate on increasingly stale groupings as training progresses. Languages that should be trained together change as parameters evolve, and clusters derived at initialization can become suboptimal or actively harmful by mid-training.
2. *Training order.* The compatibility between the current parameter state and a candidate language family changes at every regime boundary. A fixed curriculum cannot account for this: a family that represents a smooth transition at one point in training may introduce gradient conflict at another. The next family to train should be selected based on the current optimization geometry, not a predetermined schedule.
3. *Budget allocation.* Estimates of language utility derived from gradient interactions (which languages transfer well, which are still learning fast, which have plateaued) become outdated as training evolves. Static sampling ratios, and even adaptive methods that recompute ratios without re-estimating groupings, risk allocating budget based on stale utility signals.

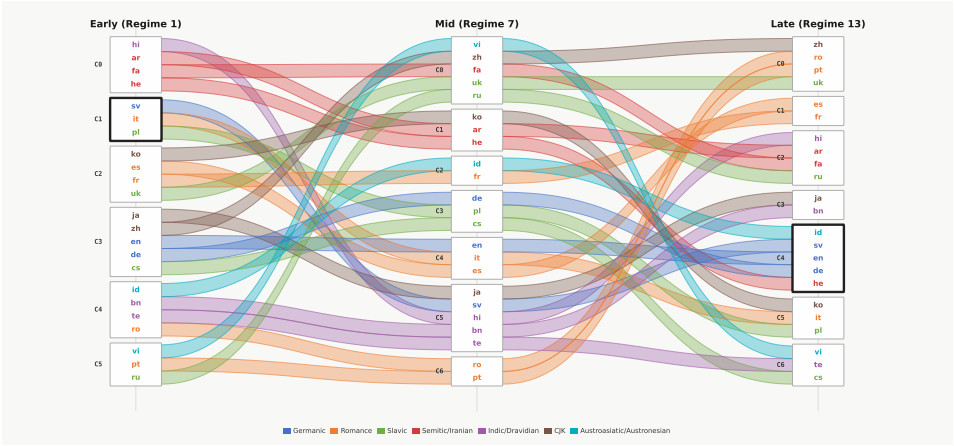

Figure 1: Language cluster migration across three training regimes (Early: Regime 1, Mid: Regime 7, Late: Regime 13) on Qwen3-4B. Ribbons trace each language's cluster trajectory, colored by linguistic family. Boxes indicate the active training family for that regime.

4 METHODOLOGY

AMIGO trains the model in discrete *regimes*, indexed by $r \in \{1, 2, \dots\}$. Each regime corresponds to a phase of training during which the cross-lingual gradient structure is assumed to be approximately stable. At each regime boundary, AMIGO re-estimates the gradient geometry, selects the next language family to train, allocates the available budget within that family, and trains until either the budget is exhausted or the gradient structure has shifted sufficiently to warrant a new regime. The procedure is summarized in Algorithm 1; the individual components are described in the subsections below.

4.1 GRADIENT ESTIMATION

Let $\mathcal{L}(\theta; x, y)$ denote the per-sample loss and $\theta \in \mathbb{R}^d$ the shared model parameters. At the start of regime r, we compute a validation gradient for each language $\ell \in \{1, \dots, L\}$:

$$g_\ell(\theta^{(r)}) \;=\; \mathbb{E}_{(x,y)\sim\mathcal{V}_\ell}\big[\nabla_\theta \mathcal{L}(\theta^{(r)}; x, y)\big], \tag{3}$$

where $\mathcal{V}_\ell$ is a held-out validation set for language ℓ. In practice, $g_\ell(\theta^{(r)})$ is approximated using a small number of mini-batches (Wang et al., 2023).

We compute these gradients only for the MLP parameters of the final transformer layer. In decoder-only LLMs, language-specific neurons concentrate in the top and bottom layers (Tang et al., 2024), and in Llama-2 the last layers move hidden states from an English-leaning latent space to the output language (Wendler et al., 2024). MLP layers act as key-value memories whose values induce distributions over the output vocabulary (Geva et al., 2021), so the final MLP is well placed to reflect that language-specific output. The restriction also bounds cost: for Qwen3-4B the final-layer MLP holds 75M of 4.0B parameters (under 2%), and computing influence or gradient similarity on a parameter subset is common practice for large models (Koh & Liang, 2017; Xia et al., 2024).

From these gradients we construct the pairwise cosine similarity matrix $S(\theta^{(r)}) \in \mathbb{R}^{L\times L}$: $S_{\ell m}(\theta) = \frac{g_\ell(\theta)^\top g_m(\theta)}{\|g_\ell(\theta)\|\,\|g_m(\theta)\|}$, which serves as the shared input to all three downstream components: language clustering, family selection, and budget allocation. All three use the same gradient pass, adding no additional forward or backward passes beyond what drift detection already requires.

4.2 REGIME DRIFT DETECTION

As θ evolves, pairwise gradient alignments between languages shift, indicating that cross-lingual relationships and optimization trade-offs are changing. We refer to this as *Pareto drift* and use it to adaptively terminate a regime.

Let $S^{(0)} = S(\theta^{(0)})$ be the baseline similarity matrix at the start of regime r, and $S^{(t)} = S(\theta^{(t)})$ its value after t steps. We monitor the relative drift

$$\rho^{(t)} = \frac{\|S^{(t)} - S^{(0)}\|_F}{\|S^{(0)}\|_F + \varepsilon}, \tag{4}$$

and trigger a regime switch when $\rho^{(t)} > \tau_{\text{rel}}$, at which point gradient-based clustering and budget allocation are recomputed for the next regime.

4.3 Hierarchical Gradient-Based Language Clustering

The Taylor expansion in Eq. (1) directly motivates grouping languages by gradient alignment: within a well-aligned family, every language benefits from updates driven by any other member, while mixing conflicting gradients causes the positive cross-lingual terms to become negative. We apply agglomerative hierarchical clustering with average linkage to the cosine dissimilarity matrix $d_{\ell m}(\theta) = 1 - S_{\ell m}(\theta)$, with K determined automatically at each regime boundary via a relative gap statistic on the dendrogram merge sequence (see Appendix A for method justification and K selection procedure).

4.4 Family Selection via Combined Compatibility Score

After each regime, AMIGO selects the next language family to train. Two objectives compete in this decision: the selected family should (i) reduce the global validation loss as much as possible, the *optimization* objective, and (ii) be compatible with the current parameter state so that the transition does not destabilize training, the *smoothness* objective. We formalize both as scalar scores and combine them linearly. Let $\bar{g}_{\mathcal{F}_k} = \frac{1}{|\mathcal{F}_k|} \sum_{\ell \in \mathcal{F}_k} g_\ell(\theta^{(r)})$ denote the mean gradient of family $\mathcal{F}_k$.

Transition Stability. The smoothness of transitioning to $\mathcal{F}_k$ is measured by the scalar projection of its mean gradient onto the previously trained family's gradient direction:

$$T(\mathcal{F}_k) = \frac{\bar{g}_{\mathcal{F}_k} \cdot \bar{g}_{\mathcal{F}_{\text{prev}}}}{\|\bar{g}_{\mathcal{F}_{\text{prev}}}\| + \varepsilon}. \tag{5}$$

To see why T captures smoothness, note that by the same Taylor expansion as Eq. (1), a step in direction $\bar{g}_{\mathcal{F}_k}$ changes the loss of the previously trained family by:

$$\Delta L_{\mathcal{F}_{\text{prev}}} \;\approx\; -\eta \langle \bar{g}_{\mathcal{F}_{\text{prev}}}, \bar{g}_{\mathcal{F}_k} \rangle \;=\; -\eta\, T(\mathcal{F}_k) \cdot \|\bar{g}_{\mathcal{F}_{\text{prev}}}\|.$$

Maximizing T therefore minimizes the expected loss increase on the previously trained family. This is equivalent to minimizing catastrophic interference between consecutive regimes (Kirkpatrick et al., 2017), and connects to gradient conflict resolution methods that explicitly constrain updates to avoid increasing prior task losses (Sener & Koltun, 2018; Yu et al., 2020; Wang et al., 2021).

A high T means $\mathcal{F}_k$'s gradient direction is aligned with the direction the model has just been moving in, predicting a smooth parameter transition. For the first regime (no previous family), we set $T(\mathcal{F}_k) = \|\bar{g}_{\mathcal{F}_k}\|$, selecting the family with the largest gradient signal to maximize early-training progress.

First-Order Transfer Gain. The expected global loss reduction from training on $\mathcal{F}_k$ is estimated via a finite-difference approximation:

$$D(\mathcal{F}_k) = -\frac{\mathcal{L}(\theta^{(r)} - \varepsilon \bar{g}_{\mathcal{F}_k}) - \mathcal{L}(\theta^{(r)})}{\varepsilon}. \tag{6}$$

Here $\mathcal{L}(\theta^{(r)}) = \frac{1}{L} \sum_{\ell=1}^{L} L_\ell(\theta^{(r)})$ denotes the uniformly-weighted mean validation loss across all L languages, evaluated on the held-out sets $\{\mathcal{V}_\ell\}$. Higher D indicates that a step in $\mathcal{F}_k$'s gradient direction would most reduce the global validation objective. This is a first-order estimate of the optimization benefit of the next regime, directly grounded in the gradient descent objective rather than a heuristic proxy.

Remark 4.1 (Approximations of Expected Loss Reduction). Both $T(\mathcal{F}_k)$ and $D(\mathcal{F}_k)$ approximate the expected global loss reduction $\langle \nabla \mathcal{L}(\theta^{(r)}), \bar{g}_{\mathcal{F}_k} \rangle$ from training on $\mathcal{F}_k$, but with different biases. $D(\mathcal{F}_k)$ estimates this via finite difference on the full validation objective accurate but greedy, tending to select the family with the steepest current descent regardless of training history. $T(\mathcal{F}_k)$ estimates it using the previous family's gradient as a local proxy for $\nabla \mathcal{L}$ cheaper but biased toward families compatible with the most recently trained languages, which risks cycling through similar families and starving others. The combined score $C(\mathcal{F}_k)$ interpolates between these two approximations: $\alpha = 0$ reduces to pure greedy descent; $\alpha = 1$ reduces to pure transition smoothness.

Compatibility Score. Both terms are min-max normalized across candidate families to make them commensurate, then combined: $C(\mathcal{F}_k \mid \theta^{(r)}) = \alpha \cdot \tilde{T}(\mathcal{F}_k) + (1 - \alpha) \cdot \tilde{D}(\mathcal{F}_k)$, where $\alpha = 0.3$ weights optimization potential more heavily than transition smoothness. The next regime trains the family with the highest score: $\mathcal{F}_{r+1} = \arg\max_{\mathcal{F}_k} C(\mathcal{F}_k \mid \theta^{(r)})$, excluding families with exhausted per-language budgets ($U_\ell \leq 0$ for all $\ell \in \mathcal{F}_k$). D requires one perturbed forward pass per family per regime boundary; T uses already-computed mean gradients and is essentially free.

4.5 Within-Family Budget Allocation

Given the active family $\mathcal{F}_s^{(r)}$, we must decide how to distribute the regime budget $W^{(r)}$ across its member languages. We formulate this as a continuous resource allocation problem and derive a closed-form solution that can be recomputed efficiently at each regime boundary.

Language Utility Parameter a_ℓ. From Eq. (1), training on language ℓ reduces the loss of language j by an amount proportional to $\langle g_\ell(\theta), g_j(\theta) \rangle$. The utility of allocating budget to ℓ should therefore reflect how much training ℓ benefits languages that currently need improvement. We measure this over out-of-family languages, those not currently being trained, weighted by their relative need: $\tilde{L}_j = \frac{L_j(\theta)}{\sum_{k \notin \mathcal{F}_s} L_k(\theta)}, \quad j \notin \mathcal{F}_s,$

$$\text{and define: } a_\ell \;=\; \max\left(0,\; \sum_{j \notin \mathcal{F}_s} \langle g_\ell(\theta),\, g_j(\theta) \rangle \cdot \tilde{L}_j \right), \tag{7}$$

where $\langle g_\ell, g_j \rangle = S_{\ell j} \cdot \|g_\ell\| \cdot \|g_j\|$ preserves gradient magnitude information (unlike cosine similarity alone). The clamp at zero handles out-of-family pairs with negative alignment. To maintain a non-degenerate objective when gradient inner products are small in magnitude as occurs in later training stages when the loss landscape flattens, we apply a relative rescaling:

$$a_\ell \;\leftarrow\; \frac{a_\ell}{\max_{k \in \mathcal{F}_s} a_k + \varepsilon} + \varepsilon_0, \tag{8}$$

with $\varepsilon_0 = 10^{-3}$. This preserves relative ordering while ensuring the knapsack objective always receives a meaningful gradient-aligned signal. All quantities are available from the gradient extraction step at zero additional compute cost.

Remark 4.2. Within the active family $\mathcal{F}_s$, languages are grouped by gradient alignment, so within-family pairs tend to have non-negative cosine similarity $S_{\ell j}(\theta) \geq 0$ for $\ell, j \in \mathcal{F}_s$, though this is not strictly guaranteed by the clustering procedure for all pairs. Out-of-family pairs $j \notin \mathcal{F}_s$ may have negative cosine similarity with $\ell \in \mathcal{F}_s$, making the clamp at zero in Eq. (7) necessary for non-negativity of the knapsack objective. The rescaling then ensures the objective remains non-degenerate as gradient inner products diminish in magnitude during later training stages.

Language Cost Parameter c_ℓ. The cost of a training sample from language ℓ should reflect how much improvement can be expected per unit of budget spent. From Eq. (1), the self-improvement of language ℓ per gradient step is proportional to $\|g_\ell(\theta)\|^2$. A language with a large gradient norm is actively learning and cheap to improve; one with a small norm has plateaued and is expensive. Additionally, a language with high current loss L_ℓ has more headroom for improvement than one already near convergence. We define: $c'_\ell = \frac{1}{L_\ell(\theta) \cdot \|g_\ell(\theta)\|^2 + \varepsilon}$, followed by $c_\ell = \frac{c'_\ell}{\sum_{k \in \mathcal{F}_s} c'_k}$, normalized over $\mathcal{F}_s$. Both $L_\ell(\theta)$ and $\|g_\ell(\theta)\|$ are available from the gradient extraction pass, eliminating the need for separate benchmark evaluation at regime boundaries. We analyze the evolution of c_ℓ across regimes for representative language families in Appendix B.

Table 1: Results across three model families under a fixed 50K instruction-tuning budget (Bactrian-X, 23 languages, 3 seeds). **Bold** indicates best result per column per model. Full results with standard deviations are in Tables 3–5 (Appendix D).

Model	Method	MultiNLU	HellaSwag	StoryCloze	MMLU-Pro	ARC-E	ARC-C	Avg
Qwen3-4B	Base	0.335	0.072	0.137	0.121	0.157	0.101	0.154
	Uniform	0.500	0.460	0.850	0.450	0.770	0.660	0.615
	PiKE	0.601	0.511	0.918	0.547	0.875	0.768	0.703
	GREATS	0.624	0.517	0.900	0.533	0.886	0.763	0.704
	RegMix	0.411	0.289	0.573	**0.581**	0.743	0.731	0.555
	MLP-MKC+	0.388	0.270	0.611	0.564	0.741	0.731	0.551
	AMIGO	**0.674**	**0.555**	**0.935**	0.556	**0.890**	**0.789**	**0.733**
LLaMA-3.2-3B	Base	0.335	0.027	0.212	0.005	0.024	0.033	0.106
	Uniform	0.281	0.232	0.451	0.374	0.520	0.504	0.394
	PiKE	0.349	0.257	0.550	0.301	0.562	0.549	0.428
	GREATS	0.332	0.300	0.756	0.304	0.440	0.377	0.418
	RegMix	0.322	0.271	0.533	**0.440**	**0.604**	**0.590**	0.460
	MLP-MKC+	0.323	0.251	0.513	0.435	0.582	0.565	0.445
	AMIGO	**0.453**	**0.383**	**0.922**	0.418	0.569	0.432	**0.529**
Gemma-3-4B	Base	0.337	0.032	0.080	0.008	0.083	0.060	0.100
	Uniform	0.308	0.240	0.454	0.355	0.571	0.554	0.414
	PiKE	0.358	0.342	0.742	0.326	**0.662**	0.528	0.493
	GREATS	0.380	0.355	0.720	0.322	0.637	0.503	0.486
	RegMix	0.355	0.272	0.516	0.453	0.651	**0.633**	0.480
	MLP-MKC+	0.365	0.255	0.506	**0.461**	0.644	0.627	0.476
	AMIGO	**0.409**	**0.381**	**0.811**	0.442	**0.662**	0.517	**0.537**

Knapsack Formulation. We model the gain from allocating x_ℓ samples to language ℓ as $a_\ell x_\ell^\beta$ with $\beta \in (0, 1)$, capturing diminishing returns: each additional sample yields smaller marginal gain than the previous one. This concave power-law form is the standard model for resource allocation under diminishing returns (Guo et al., 2025; Chen et al., 2023), and recovers the power-law scaling relationships observed empirically between data quantity and model performance (Guo et al., 2025). The allocation problem is:

$$\max_{\{x_\ell\}} \sum_{\ell \in \mathcal{F}_s^{(r)}} a_\ell \, x_\ell^\beta \quad \text{s.t.} \quad \sum_{\ell \in \mathcal{F}_s^{(r)}} c_\ell \, x_\ell \le W^{(r)}, \delta \le x_\ell \le U_\ell, \quad \forall \ell \in \mathcal{F}_s^{(r)}, \tag{9}$$

where U_ℓ is the remaining lifetime budget for language ℓ, and $\delta = \lfloor 0.05 \cdot W^{(r)} / |\mathcal{F}_s^{(r)}| \rfloor$ is a minimum allocation floor preventing any language from receiving zero training signal within its active regime. Since x^β is concave for $\beta \in (0, 1)$ and all constraints are linear, this is a convex optimization problem. The optimal solution equalizes marginal utility per unit cost across all active languages, and in the special case $c_\ell = 1$, $U_\ell = \infty$, with no language at the floor, reduces to $x_\ell^* \propto a_\ell^{1/(1-\beta)}$, recovering power-law scaling without curve fitting (Guo et al., 2025; Chen et al., 2023). The full Lagrangian derivation and Pareto-efficiency proof are in Appendix L.

5 Experiments

5.1 Experimental Setup

Models and Training. We train the base (not instruction-tuned) checkpoints of three model families: Qwen3-4B (Yang et al., 2025), LLaMA-3.2-3B (Grattafiori et al., 2024), and Gemma-3-4B (Gemma Team, 2025). All models are trained using the Bactrian-X corpus (Li et al., 2023), a multilingual instruction-tuning dataset covering 52 languages in Alpaca format (Taori et al., 2023), restricted to 23 languages: `{en, de, sv, es, fr, it, pt, ro, cs, pl, ru, uk, bn, hi, ar, fa, he, te, id, vi, ja, zh, ko}`. To assess dataset generalization, we additionally train Qwen3-4B on the Aya dataset (Singh et al., 2024) under 100K and 200K budget settings. All methods share one training recipe: the Alpaca prompt template with the loss on response tokens only, and AdamW at a learning rate of 2×10^{-5} with 100 warmup steps. We hold out 5% of each language's data, at most 100 examples, to compute validation losses and gradients. Appendix C lists the full configuration.

Baselines. We compare AMIGO against the following methods, all trained under identical data volume and model initialization conditions: **Uniform** sampling, which distributes budget equally across all languages; **PiKE** (Li et al., 2025), which maximizes non-conflicting gradient interactions for adaptive batch-level allocation; **GREATS** (Wang et al., 2024a), which selects high-quality training batches per iteration via gradient alignment; **RegMix** (Liu et al., 2025), which uses regression-based data mixture optimization; and **MLP MKC+** (Messmer et al., 2025), a dataset and benchmark track method from NeurIPS 2025.

Evaluation. We evaluate on six benchmarks covering diverse capabilities: MultiNLU (Williams et al., 2018) for Natural Language Understanding, HellaSwag (Zellers et al., 2019) and StoryCloze (Mostafazadeh et al., 2016) for Commonsense Reasoning, MMLU-Pro (Wang et al., 2024b) for Knowledge-based QA, and ARC-E and ARC-C (Clark et al., 2018) for Analytical Reasoning. All results are averaged over 3 random seeds and reported as mean $\pm$ std.

5.2 Results

Main Results and Generalization. Table 1 reports performance under a fixed 50K instruction-tuning budget across 23 languages and three model families. AMIGO consistently outperforms all baselines across all model families on an average. On Qwen3-4B, AMIGO achieves an average score of 0.733 ± 0.015, improving over the strongest baseline (GREATS: 0.704) by 2.9 points. Gains remain consistent across LLaMA-3.2-3B (0.529 vs. 0.460) and Gemma-3-4B (0.537 vs. 0.493), indicating that improvements stem from the gradient-based allocation strategy rather than model-specific effects. AMIGO is the best method on MultiNLU, HellaSwag, and StoryCloze for all three models. The fixed-mixture baselines, RegMix and MLP-MKC+, lead on MMLU-Pro for every model and on ARC-C for LLaMA-3.2-3B and Gemma-3-4B, and in each of these cases all three methods that adapt allocation during training (PiKE, GREATS, and AMIGO) score below both of them. AMIGO trails the best MMLU-Pro score by at most 0.025. The split likely comes from the training data. Bactrian-X (Li et al., 2023) is built by machine-translating the same English instructions into each language, so moving budget between languages changes how often the model sees that content without adding new knowledge. Contamination is an unlikely cause. It would affect all methods alike, because every method starts from the same pretrained checkpoint and draws on the same training data. On Aya, whose data is written natively in each language, AMIGO is best on ARC-E and ARC-C at both budgets (Table 10). It also has the best average at both budgets and gains the most from the extra data, rising from 0.681 ± 0.032 at 100K to 0.702 ± 0.031 at 200K while no baseline improves by more than 0.005. A per-language breakdown of an Aya-200K run is in Appendix H.

Language Family Migration. Figure 1 traces how gradient-based cluster assignments evolve across training regimes. The cluster composition changes substantially between early and late training: Semitic and Iranian languages (ar, fa, he) that co-cluster in Regime 1 (C0) migrate to different families by Regime 13, while Germanic languages (de, sv) initially separated in C1 and C3 converge into a single cluster by late training. Romance languages (es, fr, it, pt, ro) similarly fragment and regroup as gradient directions shift. This reorganization is the empirical signature of the Hessian–gradient rotation in Eq. (2): as parameters update, gradient similarity between language pairs changes, and the partition that minimizes cross-lingual interference must adapt accordingly. Critically, these migrations cross linguistic boundaries, CJK and Indic languages appear in different cluster configurations across regimes, demonstrating that gradient-based grouping captures optimization-space compatibility rather than surface-level typological similarity. Static methods that fix cluster assignments at initialization operate on increasingly stale groupings, while AMIGO re-estimates the partition at each regime boundary to reflect the current gradient geometry.

Ablation Study. Table 6 reports ablation results on Qwen3-4B (Bactrian-X, 50K). Every ablation outperforms Uniform (61.50) by at least 5.9 points, so each component helps on its own. Scheduling is the strongest single component (`s_only`: 69.94), ahead of clustering (`c_only`: 67.92) and knapsack allocation (`k_only`: 67.43). Removing any one component lowers the average, in the same order: by 3.58 points without scheduling (`no_scheduling`: 69.74), 2.74 without clustering, and 1.89 without knapsack allocation. The full method (73.32) is best on average and on four of the six benchmarks. `no_knapsack` scores higher on HellaSwag and MMLU-Pro, and most of AMIGO's margin over it comes from MultiNLU.

Sensitivity Analysis. Table 7 (Appendix F) reports performance across $\alpha \in [0.0, 0.5]$, the weight controlling the balance between the stability term T and the first-order transfer gain D in family selection. Performance peaks at $\alpha = 0.30$ (avg 0.733) and degrades at both extremes: pure greedy descent ($\alpha = 0$, avg 0.714) underperforms by 1.9 points, confirming that transition smoothness provides meaningful regularization beyond optimization potential alone, while excessive smoothing ($\alpha = 0.5$, avg 0.711) sacrifices too much descent signal. AMIGO is robust to the choice of the diminishing-returns exponent β: Table 8 shows variation of only 0.015 across $\beta \in [0.1, 0.7]$, with a peak at $\beta = 0.4$. We use $\alpha = 0.30$ and $\beta = 0.4$ as defaults throughout. Table 9 further reports sensitivity to the drift threshold τ_{rel}, which governs regime frequency; performance peaks at $\tau_{\text{rel}} = 1.4$ (21 regimes) and degrades when regimes are either too frequent or too infrequent.

Computational Overhead. We report wall-clock training times on a single H100 GPU under identical conditions (Qwen3-4B, Bactrian-X, 50K budget). Uniform sampling completes in 3,208s and serves as the overhead-free baseline. AMIGO runs in 4,060s, virtually identical to PiKE at 4,029s, and substantially faster than GREATS at 11,078s. Despite performing richer per-regime decisions, gradient-based language clustering, compatibility-driven family selection, and knapsack budget allocation, AMIGO matches PiKE's wall-clock time while delivering stronger performance, and is $2.7\times$ faster than GREATS. GREATS' disproportionate cost stems from per-iteration batch selection via gradient inner-products, which scales with every training step. AMIGO's overhead, by contrast, is concentrated at its regime boundaries over the full training run, making it predictable and independent of the number of training steps. On the other hand, RegMix and MLP-MKC+ rely on proxy models or large-scale offline precomputation, which, although performed only once, introduce substantial upfront computational cost. In particular, both RegMix and MLP-MKC+ require multi-day training on multiple H100 GPUs over large token budgets.

Comparison with Model-Agnostic and Embedding-Based Methods. Model-agnostic data selection methods, such as those based on embedding similarity or proxy-model influence estimates (e.g., RegMix and MLP-MKC+), avoid repeated gradient computation and can be applied once before training begins. However, their decisions are derived from static proxies that remain fixed throughout optimization. As a result, they cannot adapt to evolving cross-lingual interactions or changing language learning dynamics as model parameters update. In contrast, AMIGO recomputes training decisions directly from the target model's current optimization state using validation gradients, gradient geometry, and loss statistics. This introduces additional overhead, but enables model- and training-state-specific grouping, scheduling, and budget allocation. Empirically, the resulting gains remain consistent across **three** architecturally distinct model families, suggesting that the gradient-based signals capture transferable optimization dynamics rather than architecture-specific behavior.

6 Conclusion

We introduced AMIGO, a regime-based framework for adaptive multilingual instruction tuning that models cross-lingual interactions through evolving gradient geometry. Using validation-gradient similarity, AMIGO dynamically clusters languages, selects compatible training families, and allocates budget through a gradient-grounded continuous optimization objective. Unlike static data-mixing or fixed curriculum approaches, AMIGO adapts training decisions throughout optimization, enabling more effective utilization of limited multilingual training budgets. Across multiple model families, datasets, and evaluation benchmarks, AMIGO improves multilingual transfer and instruction-following performance under fixed training budgets, demonstrating the effectiveness of jointly modeling grouping, scheduling, and allocation within a unified optimization framework. More broadly, our results suggest that multilingual training is inherently non-stationary, and that accounting for evolving gradient interactions provides a useful perspective for scalable multilingual optimization. While AMIGO relies on computationally efficient first-order gradient approximations and partial parameter-space estimates to remain practical at scale, the framework provides a flexible foundation for future extensions involving richer optimization signals and broader multilingual settings.

ETHICS STATEMENT

This work aims to make multilingual model training more efficient and adaptive, enabling stronger support for low-resource and underrepresented languages without requiring proportional increases in data or computation. By improving the distribution of training attention across languages, the proposed approach can help reduce performance disparities and contribute to more inclusive language technologies. As with any multilingual system, careful consideration of data quality and evaluation practices remains essential to ensure that these benefits are realized fairly across linguistic communities.

REPRODUCIBILITY STATEMENT

The components required to reproduce our results are described in the paper and appendix. The AMIGO procedure—gradient estimation, regime drift detection, language clustering, family selection, and budget allocation—is specified in Section 4, with full pseudo-code in Algorithm 1 and additional clustering details in Appendix A. The derivation and solution of the continuous knapsack objective are given in Appendix L. Section 5 describes the models, datasets, training budgets, baselines, and evaluation benchmarks; per-benchmark results with standard deviations are reported in Appendix D, ablations in Appendix E, and hyperparameter sensitivity in Appendix F. All models and datasets used in this work are publicly available, and their licenses are listed in Appendix M.

AI USE STATEMENT

We used generative AI tools for writing assistance, specifically for copy-editing prose and for suggesting revisions to the organization of the manuscript, and as a coding assistant for analysis and plotting scripts used to produce the tables and figures. All AI-assisted text was reviewed and edited by the authors, all reported numbers were generated from the experimental artifacts and checked against the analysis code, and any AI-assisted code was read and tested by the authors. We take responsibility for the final content of this work, including all text, claims, and artifacts.

REFERENCES

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S. Morcos. SemDeDup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*, 2023.

Liang Chen, Shuming Ma, Dongdong Zhang, Furu Wei, and Baobao Chang. On the Pareto front of multilingual neural machine translation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 33188–33201. Curran Associates, Inc., 2023.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Vincent Cohen-Addad, Varun Kanade, Frederik Mallmann-Trenn, and Claire Mathieu. Hierarchical clustering: Objective functions and algorithms. *Journal of the ACM*, 66(4):26:1–26:42, 2019.

Simin Fan, Matteo Pagliardini, and Martin Jaggi. DOGE: Domain reweighting with generalization estimation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12895–12915, 2024.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27503–27516. Curran Associates, Inc., 2021.

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495. Association for Computational Linguistics, 2021.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ping Guo, Yubing Ren, Binbin Liu, Fengze Liu, Haobin Lin, Yifan Zhang, Bingni Zhang, Taifeng Wang, and Yin Zheng. Exploring polyglot harmony: On multilingual data allocation for large language models pretraining. *arXiv preprint arXiv:2509.15556*, 2025.

Pritam Kadasi, Abhishek Upperwal, and Mayank Singh. ADAPT: Learning task mixtures for budget-constrained instruction tuning. *arXiv preprint arXiv:2512.04555*, 2025.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. AutoScale: Scale-aware data mixing for pre-training LLMs. In *Conference on Language Modeling*, 2025.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894, 2017.

Hugo Laurençon et al. The BigScience ROOTS corpus: A 1.6TB composite multilingual dataset. In *Advances in Neural Information Processing Systems*, volume 35, pp. 31809–31826. Curran Associates, Inc., 2022.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8424–8445, Dublin, Ireland, 2022. Association for Computational Linguistics.

Haonan Li, Fajri Koto, Minghao Wu, Alham Fikri Aji, and Timothy Baldwin. Bactrian-X: Multilingual replicable instruction-following models with low-rank adaptation. *arXiv preprint arXiv:2305.15011*, 2023.

Zeman Li, Yuan Deng, Peilin Zhong, Meisam Razaviyayn, and Vahab Mirrokni. PiKE: Adaptive data mixing for large-scale multi-task learning under low gradient conflicts. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. Not all tokens are what you need for pretraining. In *Advances in Neural Information Processing Systems*, volume 37, pp. 29029–29063. Curran Associates, Inc., 2024.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. RegMix: Data mixture as regression for language model pre-training. In *International Conference on Learning Representations*, 2025.

Bettina Messmer, Vinko Sabolčec, and Martin Jaggi. Enhancing multilingual LLM pretraining with model-based data selection. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 839–849, San Diego, California, 2016. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Haebin Shin, Lei Ji, Xiao Liu, Zhiwei Yu, Hyunwoo Yoo, Qi Chen, and Yeyun Gong. DynamixSFT: Dynamic mixture optimization of instruction tuning collections. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 39590–39603, San Diego, California, United States, 2026. Association for Computational Linguistics.

Shivalika Singh et al. Aya dataset: An open-access collection for multilingual instruction tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11521–11567, Bangkok, Thailand, 2024. Association for Computational Linguistics.

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. Language-specific neurons: The key to multilingual capabilities in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5701–5715, Bangkok, Thailand, 2024. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford Alpaca: An instruction-following LLaMA model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

Jiachen T. Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. GREATS: Online selection of high-quality data for LLM training in every iteration. In *Advances in Neural Information Processing Systems*, volume 37, pp. 131197–131223. Curran Associates, Inc., 2024a.

Mingyang Wang, Heike Adel, Lukas Lange, Jannik Strötgen, and Hinrich Schuetze. GradSim: Gradient-based language grouping for effective multilingual training. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4631–4646, Singapore, 2023. Association for Computational Linguistics.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024b.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*, 2021.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. Do Llamas work in English? On the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15366–15394, Bangkok, Thailand, 2024. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132, 2024.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Haneul Yoo, Cheonbok Park, Sangdoo Yun, Alice Oh, and Hwaran Lee. Code-switching curriculum learning for multilingual transfer in LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 7816–7836, Vienna, Austria, 2025. Association for Computational Linguistics.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5824–5836. Curran Associates, Inc., 2020.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, 2019. Association for Computational Linguistics.

Mingliang Zhang, Fandong Meng, Yunhai Tong, and Jie Zhou. Competence-based curriculum learning for multilingual machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 2481–2493, Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics.

A Clustering Method Details

The Taylor expansion in Eq. (1) shows that training on language ℓ reduces the loss of language i by an amount proportional to $\langle \nabla L_i(\theta), \nabla L_\ell(\theta) \rangle$. This directly motivates training languages together when their gradients are aligned: within a well-aligned group, every language simultaneously benefits from updates driven by any other member. Conversely, mixing languages with conflicting gradients causes the positive cross-lingual terms in Eq. (1) to become negative, inducing interference rather than transfer. Language grouping is therefore not a heuristic preprocessing step but a direct consequence of the optimization objective: a good partition is one that maximizes within-group gradient alignment and minimizes cross-group interference.

Dissimilarity measure. We convert the pairwise cosine similarity $S_{\ell m}(\theta)$ into a metric distance:

$$d_{\ell m}(\theta) = 1 - S_{\ell m}(\theta). \tag{10}$$

Smaller $d_{\ell m}(\theta)$ indicates stronger gradient alignment; larger values correspond to conflicting optimization directions. The induced distance matrix $D(\theta) = [d_{\ell m}(\theta)]$ captures the full cross-lingual compatibility structure at the current parameter state.

Clustering method. We apply agglomerative hierarchical clustering (Cohen-Addad et al., 2019) with average linkage to $D(\theta)$, producing a dendrogram over languages. Three properties make this the natural choice for our setting. First, it operates directly on $D(\theta)$ without assuming a parametric cluster shape, making it robust when gradient similarity patterns change unpredictably across regimes. Second, it does not require specifying K in advance: the dendrogram encodes all possible partitions simultaneously, and we extract the one that best fits the current gradient structure. Third, it preserves the nested structure of cross-lingual relationships, languages that are close in gradient space at coarse granularity remain together as clusters are refined, which ensures that the family decomposition is consistent with the optimization geometry at multiple scales.

Automatic determination of K. The number of families K is determined at each regime boundary by analyzing the merge distance sequence of the dendrogram. Let $\delta_1 \leq \delta_2 \leq \cdots \leq \delta_{L-1}$ denote the distances at which successive merges occur. We identify K as the number of clusters at the step where the *relative gap*

$$\gamma_k = \frac{\delta_{k+1} - \delta_k}{\delta_k + \varepsilon} \tag{11}$$

is maximized, corresponding to the largest proportional jump in merge distance. This gap statistic has a natural interpretation: a large jump at step k means that the $L - k$ remaining clusters are substantially more internally coherent than any coarser partition, and merging them would combine groups that are genuinely incompatible in gradient space. Since gradient geometry changes across regimes, K is recomputed at each regime boundary rather than fixed globally, allowing the number of families to adapt to the current optimization landscape. Across training regimes, K ranges from 3 to 7, demonstrating that the gap statistic adapts the number of families to the current gradient geometry rather than fixing it globally.

B Evolution of Cost Parameter c_ℓ Across Regimes

Figure 2 tracks the cost parameter c_ℓ for representative languages within four language families : Indic (hi, te, bn), CJK (ja, zh, ko), Slavic (ru, uk, cs, pl), and European (de, es, it, fr) across all 51 training regimes on Qwen3-4B (Bactrian-X, 50K budget). Recall that c_ℓ is defined as the normalized inverse of the product of validation loss and squared gradient norm, reflecting the per-sample cost of improving language ℓ: a high c_ℓ indicates that the language has either plateaued (small gradient norm) or is inherently difficult to improve per unit of budget.

Three patterns are apparent. First, Indic languages (hi, te, bn) consistently exhibit the highest c_ℓ values ($\sim$0.45–0.60) across all regimes. This reflects their early convergence: Hindi and Telugu reach low validation losses quickly, causing their gradient norms to shrink and making further improvement expensive per training sample. The knapsack formulation consequently deprioritizes these languages in later regimes relative to their nominal loss, correctly redirecting budget toward languages where

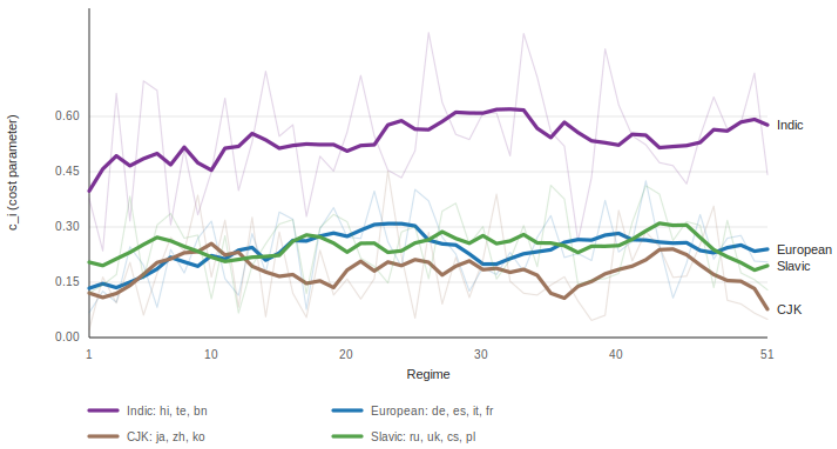

Figure 2: Mean c_ℓ (cost parameter) per language family across 51 training regimes. Bold lines show group averages; faint traces show individual languages within each group. Indic languages (hi, te, bn) incur the highest and most stable cost throughout training, while CJK languages (ja, zh, ko) remain cheapest. European and Slavic groups occupy an intermediate and gradually converging range.

marginal gains are larger. Second, CJK languages (ja, zh, ko) maintain the lowest c_ℓ values ($\sim$0.10–0.25) throughout training, driven by persistently large gradient norms, a signal that these languages continue to learn actively and remain cheap to improve per sample. This is consistent with their sustained high validation losses observed in the main experiments, and explains why the knapsack allocates them non-trivial budget even in late regimes. Third, European and Slavic groups occupy a stable intermediate range ($\sim$0.20–0.30), with modest upward drift over regimes as these languages gradually approach convergence. The between-group separation is maintained throughout training, confirming that c_ℓ reliably stratifies languages by their learning speed without requiring any external benchmark evaluation at regime boundaries.

C Implementation Details

Table 2 lists the configuration used for every experiment. The upper block is shared by all methods; the lower block applies to AMIGO only.

D Full Results with Standard Deviations

Tables 3, 4, and 5 report per-task benchmark scores across 3 random seeds (42, 43, 44) for Qwen3-4B, LLaMA-3.2-3B, and Gemma-3-4B respectively, under the 50K Bactrian-X budget.

Table 2: Training and evaluation configuration.

All methods	
Base checkpoints	`Qwen/Qwen3-4B-Base`, `meta-llama/Llama-3.2-3B`, `google/gemma-3-4b-pt`
Training data	Bactrian-X, 23 languages, up to 12K examples per language
Training budget	50K samples; Aya experiments use 100K and 200K (Qwen3-4B only)
Prompt format	Alpaca template; loss on response tokens only
Validation split	5% of each language's data, at most 100 examples
Maximum sequence length	512 tokens
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Learning rate	2×10^{-5}, linear schedule, 100 warmup steps
Precision	bf16 with gradient checkpointing
Hardware	one NVIDIA H100 GPU
Evaluation languages	ar, bn, de, en, fr, hi, ja, ru, sv, zh
AMIGO only	
Gradient parameters	MLP of the final transformer layer
Validation gradients	cross-entropy on the held-out split, recomputed at each regime boundary
Regime length	at most 500 optimizer steps, ending early when $\rho^{(t)} > \tau_{\text{rel}}$
Drift threshold τ_{rel}	1.4
Compatibility weight α	0.3
Diminishing-returns exponent β	0.4
Finite-difference step ε in Eq. (6)	10^{-4}
Allocation floor δ	$\lfloor 0.05\, W^{(r)}/\lvert\mathcal{F}_s^{(r)}\rvert \rfloor$
Clustering	agglomerative, average linkage; K at the largest relative merge gap
Knapsack solver	closed form (Appendix L); λ by binary search

Table 3: Full results with std for **Qwen3-4B**, Bactrian-X 50K. **Bold** = best per column.

Method	MultiNLU	HellaSwag	StoryCloze	MMLU-Pro	ARC-E	ARC-C	Avg
Base	0.335 ± 0.005	0.072 ± 0.003	0.137 ± 0.004	0.121 ± 0.002	0.157 ± 0.004	0.101 ± 0.003	0.154 ± 0.004
Uniform	0.500 ± 0.020	0.460 ± 0.021	0.850 ± 0.016	0.450 ± 0.027	0.770 ± 0.024	0.660 ± 0.019	0.615 ± 0.021
PiKE	0.601 ± 0.028	0.511 ± 0.038	0.918 ± 0.024	0.547 ± 0.009	0.875 ± 0.032	0.768 ± 0.029	0.703 ± 0.027
GREATS	0.624 ± 0.010	0.517 ± 0.031	0.900 ± 0.024	0.533 ± 0.018	0.886 ± 0.026	0.763 ± 0.012	0.704 ± 0.020
RegMix	0.411 ± 0.019	0.289 ± 0.033	0.573 ± 0.016	$\mathbf{0.581 \pm 0.028}$	0.743 ± 0.023	0.731 ± 0.022	0.555 ± 0.023
MLP-MKC+	0.388 ± 0.016	0.270 ± 0.018	0.611 ± 0.016	0.564 ± 0.016	0.741 ± 0.017	0.731 ± 0.016	0.551 ± 0.017
AMIGO	$\mathbf{0.674 \pm 0.012}$	$\mathbf{0.555 \pm 0.014}$	$\mathbf{0.935 \pm 0.015}$	0.556 ± 0.018	$\mathbf{0.890 \pm 0.013}$	$\mathbf{0.789 \pm 0.017}$	$\mathbf{0.733 \pm 0.015}$

E Ablation Study

Table 6: Ablation study on Qwen3-4B, Bactrian-X 50K. `c_only`: clustering only; `k_only`: knapsack allocation only; `s_only`: scheduling only; `no_clustering`, `no_knapsack`, `no_scheduling`: full AMIGO with the respective component removed. **Bold** = best per row.

Dataset	Uniform	c_only	k_only	s_only	no_clustering	no_knapsack	no_scheduling	AMIGO
MultiNLU	50.00	57.63	55.18	58.27	57.46	58.13	56.84	**67.40**
ARC-E	77.00	84.37	83.92	85.86	86.74	87.31	85.47	**89.00**
ARC-C	66.00	72.64	71.83	73.91	74.86	75.38	73.47	**78.90**
HellaSwag	46.00	52.37	53.61	55.84	57.28	**58.43**	56.37	55.50
MMLU-Pro	45.00	51.74	50.63	54.87	55.91	**57.16**	55.38	55.60
StoryCloze	85.00	88.76	89.43	90.87	91.24	92.18	90.93	**93.50**
Average	61.50	67.92	67.43	69.94	70.58	71.43	69.74	**73.32**

Table 4: Full results with std for **LLaMA-3.2-3B**, Bactrian-X 50K. **Bold** = best per column.

Method	MultiNLU	HellaSwag	StoryCloze	MMLU-Pro	ARC-E	ARC-C	Avg
Base	0.335 ± 0.005	0.027 ± 0.002	0.212 ± 0.005	0.005 ± 0.001	0.024 ± 0.002	0.033 ± 0.002	0.106 ± 0.003
Uniform	0.281 ± 0.022	0.232 ± 0.021	0.451 ± 0.020	0.374 ± 0.028	0.520 ± 0.026	0.504 ± 0.021	0.394 ± 0.023
PiKE	0.349 ± 0.016	0.257 ± 0.019	0.550 ± 0.026	0.301 ± 0.007	0.562 ± 0.095	0.549 ± 0.067	0.428 ± 0.038
GREATS	0.332 ± 0.015	0.300 ± 0.022	0.756 ± 0.036	0.304 ± 0.004	0.440 ± 0.016	0.377 ± 0.014	0.418 ± 0.018
RegMix	0.322 ± 0.030	0.271 ± 0.026	0.533 ± 0.019	$\mathbf{0.440 \pm 0.034}$	$\mathbf{0.604 \pm 0.032}$	$\mathbf{0.590 \pm 0.026}$	0.460 ± 0.028
MLP-MKC+	0.323 ± 0.023	0.251 ± 0.016	0.513 ± 0.026	0.435 ± 0.018	0.582 ± 0.014	0.565 ± 0.013	0.445 ± 0.018
AMIGO	$\mathbf{0.453 \pm 0.017}$	$\mathbf{0.383 \pm 0.012}$	$\mathbf{0.922 \pm 0.017}$	0.418 ± 0.010	0.569 ± 0.014	0.432 ± 0.016	$\mathbf{0.529 \pm 0.014}$

Table 5: Full results with std for **Gemma-3-4B**, Bactrian-X 50K. **Bold** = best per column.

Method	MultiNLU	HellaSwag	StoryCloze	MMLU-Pro	ARC-E	ARC-C	Avg
Base	0.337 ± 0.005	0.032 ± 0.002	0.080 ± 0.003	0.008 ± 0.001	0.083 ± 0.003	0.060 ± 0.002	0.100 ± 0.003
Uniform	0.308 ± 0.019	0.240 ± 0.024	0.454 ± 0.012	0.355 ± 0.027	0.571 ± 0.022	0.554 ± 0.020	0.414 ± 0.021
PiKE	0.358 ± 0.023	0.342 ± 0.080	0.742 ± 0.015	0.326 ± 0.019	$\mathbf{0.662 \pm 0.017}$	0.528 ± 0.012	0.493 ± 0.028
GREATS	0.380 ± 0.017	0.355 ± 0.015	0.720 ± 0.025	0.322 ± 0.012	0.637 ± 0.014	0.503 ± 0.015	0.486 ± 0.016
RegMix	0.355 ± 0.025	0.272 ± 0.028	0.516 ± 0.014	0.453 ± 0.029	0.651 ± 0.030	$\mathbf{0.633 \pm 0.026}$	0.480 ± 0.025
MLP-MKC+	0.365 ± 0.014	0.255 ± 0.019	0.506 ± 0.025	$\mathbf{0.461 \pm 0.018}$	0.644 ± 0.025	0.627 ± 0.026	0.476 ± 0.021
AMIGO	$\mathbf{0.409 \pm 0.019}$	$\mathbf{0.381 \pm 0.017}$	$\mathbf{0.811 \pm 0.010}$	0.442 ± 0.011	$\mathbf{0.662 \pm 0.017}$	0.517 ± 0.018	$\mathbf{0.537 \pm 0.015}$

F Sensitivity Analysis

AMIGO has three key hyperparameters: the compatibility weight α, the diminishing-returns exponent β and the drift threshold τ_{rel}. Tables 7–9 report average benchmark scores across their respective ranges on Qwen3-4B, Bactrian-X, 50K budget.

Compatibility weight α. Table 7 shows performance across $\alpha \in [0.0, 0.5]$. $\alpha = 0$ corresponds to pure finite-difference (greedy global descent); $\alpha = 1$ would be pure transition smoothness. Performance peaks at $\alpha = 0.30$ (avg 0.733), confirming that weighting optimization potential more heavily than transition smoothness is beneficial, while a small stability component prevents abrupt transitions. Pure greedy selection ($\alpha = 0$, avg 0.714) underperforms, validating the need for the transition term.

Table 7: Sensitivity to compatibility weight α. $\alpha = 0.30$ is used as default.

α	Avg
0.00	0.714
0.05	0.719
0.10	0.724
0.30	**0.733**
0.50	0.711

Diminishing-returns exponent β. Table 8 reports performance across $\beta \in [0.1, 0.7]$. AMIGO is robust to this choice, with variation of only 0.015 across the full range. Performance peaks at $\beta = 0.4$ (avg 0.733) and degrades symmetrically toward both extremes: very small β approaches a logarithmic utility model that over-compresses differences in language utility, while large β approaches linear allocation that ignores diminishing returns. We use $\beta = 0.4$ as default.

Drift threshold τ_{rel}. Table 9 reports performance across $\tau_{\text{rel}} \in \{1.2, 1.4, 1.6\}$, which controls how sensitively AMIGO detects gradient drift and triggers a regime switch. A lower τ_{rel} induces more frequent regime updates (66 regimes at 1.2 vs. 13 at 1.6), increasing adaptivity at the cost of overhead. Performance peaks at $\tau_{\text{rel}} = 1.4$ (avg 0.732, 21 regimes) and degrades in both directions: $\tau_{\text{rel}} = 1.2$ over-segments the training run into too many short regimes, reducing the time the model spends in each gradient geometry before re-clustering; $\tau_{\text{rel}} = 1.6$ under-detects drift, causing AMIGO to operate on stale gradient structure for too long. We use $\tau_{\text{rel}} = 1.4$ as the default throughout.

Table 8: Sensitivity to diminishing-returns exponent β.

β	**Avg Score**
0.1	0.728
0.3	0.725
0.4	**0.733**
0.5	0.727
0.7	0.718

Table 9: Sensitivity to drift threshold τ_{rel}. $\tau_{\text{rel}} = 1.4$ is used as default.

τ_{rel}	**Avg Score**	**No. of Regimes**
1.2	0.709	66
1.4	**0.732**	21
1.6	0.727	13

G Generalization Beyond Bactrian-X

To assess the generalizability of AMIGO beyond the Bactrian-X dataset, we conducted additional experiments using the Aya dataset under two different budget settings: 100K and 200K samples. The results, summarized in Table 10, align with the observations made on the Bactrian-X dataset in the main text. AMIGO consistently maintains or exceeds the performance of state-of-the-art baselines, demonstrating robust cross-lingual transfer without suffering from negative interference even on different data distributions. We plan to conduct a deeper gradient-based behavior analysis on these two types of datasets in the final manuscript.

Table 10: Model performance generalizing on Aya dataset under two budget settings (100k and 200k samples) across different baselines.

Model	**Method**	**MultiNLU**	**HellaSwag**	**StoryCloze**	**MMLU-Pro**	**ARC-E**	**ARC-C**	**Avg**
Qwen (200k)	**AMIGO**	0.718 ± 0.050	$\mathbf{0.560 \pm 0.030}$	$\mathbf{0.945 \pm 0.021}$	$\mathbf{0.282 \pm 0.036}$	$\mathbf{0.926 \pm 0.020}$	$\mathbf{0.783 \pm 0.030}$	$\mathbf{0.702 \pm 0.031}$
	PiKE	0.668 ± 0.055	0.554 ± 0.024	0.882 ± 0.015	0.253 ± 0.039	0.881 ± 0.021	0.751 ± 0.035	0.665 ± 0.032
	GREATS	0.664 ± 0.058	0.548 ± 0.021	0.901 ± 0.013	0.250 ± 0.034	0.879 ± 0.021	0.750 ± 0.038	0.665 ± 0.031
	RegMix	0.710 ± 0.031	0.514 ± 0.035	0.922 ± 0.018	0.242 ± 0.022	0.895 ± 0.019	0.754 ± 0.033	0.673 ± 0.026
	MLP MKC+	$\mathbf{0.720 \pm 0.035}$	0.508 ± 0.041	0.919 ± 0.019	0.251 ± 0.025	0.888 ± 0.017	0.762 ± 0.029	0.675 ± 0.028
Qwen (100k)	**AMIGO**	0.699 ± 0.037	$\mathbf{0.554 \pm 0.041}$	0.911 ± 0.029	0.233 ± 0.031	$\mathbf{0.916 \pm 0.025}$	$\mathbf{0.771 \pm 0.032}$	$\mathbf{0.681 \pm 0.032}$
	PiKE	0.674 ± 0.054	0.552 ± 0.025	0.884 ± 0.014	0.252 ± 0.034	0.883 ± 0.022	0.750 ± 0.035	0.666 ± 0.031
	GREATS	0.664 ± 0.057	0.546 ± 0.023	0.900 ± 0.015	0.252 ± 0.037	0.880 ± 0.022	0.750 ± 0.038	0.665 ± 0.032
	RegMix	0.680 ± 0.035	0.517 ± 0.040	$\mathbf{0.918 \pm 0.019}$	$\mathbf{0.256 \pm 0.025}$	0.893 ± 0.018	0.755 ± 0.035	0.670 ± 0.029
	MLP MKC+	$\mathbf{0.711 \pm 0.042}$	0.503 ± 0.034	0.914 ± 0.018	0.252 ± 0.026	0.889 ± 0.017	0.751 ± 0.038	0.670 ± 0.029

H Per-Language Allocation on Aya

Table 11 breaks down AMIGO's Aya-200K run on Qwen3-4B ($\alpha = 0.05$) by language. For each language it lists the number of regimes in which the language was trained, the samples it received, and the relative drop in its mean training loss between the first and the last of those regimes. The budget reaches all 23 languages. The largest share goes to Chinese (10.4%), and 13 languages receive less than a uniform share (4.35%). Training loss falls for 22 of the 23 languages; German rises by 2.8%. The drop is only weakly related to allocation (Spearman $\rho = 0.36$): Swedish and Arabic receive less than a uniform share yet show the two largest drops (25.7% and 23.5%), well above English (9.0%) and Chinese (10.1%). These are training losses on each language's own batches, so they show where optimization made progress rather than held-out performance, which Table 10 reports.

Table 11: Per-language allocation and training-loss change for AMIGO on Aya-200K (Qwen3-4B, $\alpha = 0.05$). Loss drop is the relative decrease in mean training loss between the first and last regime in which the language was trained; negative values are increases.

Language	Regimes	Samples	Share (%)	Loss drop (%)
zh	11	20,816	10.4	10.10
he	9	17,960	9.0	7.98
ko	8	16,112	8.1	6.22
it	12	14,664	7.3	13.38
vi	8	13,376	6.7	4.59
id	11	12,328	6.2	11.40
ja	10	11,480	5.7	9.63
ru	11	11,112	5.6	18.75
en	12	9,744	4.9	8.98
te	8	9,632	4.8	14.84
ar	7	8,408	4.2	23.52
pl	11	7,016	3.5	7.60
es	8	6,400	3.2	4.28
fr	5	6,064	3.0	7.70
ro	3	5,704	2.9	4.93
sv	7	5,248	2.6	25.72
de	7	4,912	2.5	−2.81
pt	5	4,808	2.4	7.98
bn	8	4,640	2.3	10.46
hi	6	4,128	2.1	10.84
cs	9	2,528	1.3	2.79
uk	8	2,096	1.0	1.60
fa	5	824	0.4	3.61

I Detailed Pseudo-codes of Proposed Methodology

AMIGO Algorithm

The AMIGO algorithm implements a regime-based multilingual training strategy that adaptively allocates training resources based on the evolving geometry of language-specific gradients. At the beginning of each regime, AMIGO computes validation-gradient similarities across languages to (i) estimate cross-lingual transfer potential and (ii) cluster languages into families with aligned optimization behavior. Given these signals, AMIGO solves a per-regime knapsack-style optimization to allocate a finite training budget across languages. Training then proceeds within a selected language family, using allocation-proportional sampling, until significant gradient drift or budget exhaustion triggers a regime transition. Algorithms 1–3 detail the main training loop, gradient-based utility estimation, and compatibility-driven family selection used throughout this process.

J Effect of Gradient Alignment on Per-Language Loss

Let $\theta \in \mathbb{R}^d$ denote the shared model parameters. Each language $\ell \in [L]$ induces a loss function $\mathcal{L}_\ell(\theta)$. Multilingual instruction tuning optimizes a weighted objective

$$\mathcal{L}(\theta) = \sum_{\ell=1}^{L} w_\ell \mathcal{L}_\ell(\theta), \tag{12}$$

where $w_\ell \geq 0$ are language-specific weights satisfying $\sum_{\ell=1}^{L} w_\ell = 1$.

A single gradient descent step with learning rate $\eta > 0$ updates parameters as

$$\theta^+ = \theta - \eta \sum_{\ell=1}^{L} w_\ell \nabla_\theta \mathcal{L}_\ell(\theta). \tag{13}$$

Algorithm 1 AMIGO: Main Training Loop

1: Initialize regime counter $r \leftarrow 0$
2: **while** training not complete **do**
3: **Gradient Analysis and Clustering:** Compute validation gradients for all L languages and cluster into families:
$$\{\mathcal{F}_1, \ldots, \mathcal{F}_K\} \leftarrow \text{Algorithm 2.}$$
4: **Next Family Selection:** Evaluate compatibility scores $C(\mathcal{F}_k \mid \theta^{(r)})$ for all families using Algorithm 3, and select
$$\mathcal{F}_s^{(r)} \leftarrow \arg\max_{\mathcal{F}_k} C(\mathcal{F}_k \mid \theta^{(r)}).$$
5: **Regime Budget Allocation (Knapsack):** Compute $\{a_i^{(r)}, c_i^{(r)}\}_{i \in \mathcal{F}_s^{(r)}}$ from Algorithm 2 restricted to $\mathcal{F}_s^{(r)}$, then solve
$$\{x_i^{(r)}\}_{i \in \mathcal{F}_s^{(r)}} \leftarrow \arg\max_{\{x_i\}} \sum_{i \in \mathcal{F}_s^{(r)}} a_i^{(r)} x_i^{\beta} \quad \text{s.t.} \sum_{i \in \mathcal{F}_s^{(r)}} c_i^{(r)} x_i \le W^{(r)},\ \delta \le x_i \le U_i.$$
6: **for** each training batch within family $\mathcal{F}_s^{(r)}$ **do**
7: **Within-Family Sampling:** For each language $\ell \in \mathcal{F}_s^{(r)}$, define the sampling proportion
$$p_\ell = \frac{x_\ell^{(r)}}{\sum_{\ell' \in \mathcal{F}_s^{(r)}} x_{\ell'}^{(r)}}.$$
8: Train the model on languages in $\mathcal{F}_s^{(r)}$ using sampling ratios $\{p_\ell\}_{\ell \in \mathcal{F}_s^{(r)}}$.
9: **end for**
10: **Regime Drift Check:** Monitor gradient drift $\rho^{(t)}$ within the regime.
11: **if** $\rho^{(t)} > \tau_{\text{rel}}$ **or** any language exhausts its allocated budget **then**
12: Update remaining budget $W^{(r+1)}$, set $\theta^{(r+1)} \leftarrow \theta$, and increment regime counter $r \leftarrow r+1$
13: **else**
14: Continue training within the current regime
15: **end if**
16: **end while**

We analyze the change in loss for a particular language $i \in [L]$:

$$\Delta\mathcal{L}_i = \mathcal{L}_i(\theta^+) - \mathcal{L}_i(\theta). \tag{14}$$

Using a first-order Taylor expansion around θ,

$$\mathcal{L}_i(\theta^+) \approx \mathcal{L}_i(\theta) + \nabla_\theta \mathcal{L}_i(\theta)^\top (\theta^+ - \theta). \tag{15}$$

Substituting the update rule yields

$$\Delta\mathcal{L}_i \approx -\eta \sum_{\ell=1}^{L} w_\ell \, \nabla_\theta \mathcal{L}_i(\theta)^\top \nabla_\theta \mathcal{L}_\ell(\theta) \tag{16}$$

$$= -\eta w_i \|\nabla_\theta \mathcal{L}_i(\theta)\|_2^2 - \eta \sum_{\ell \neq i} w_\ell \langle \nabla_\theta \mathcal{L}_i(\theta), \nabla_\theta \mathcal{L}_\ell(\theta) \rangle. \tag{17}$$

The first term is non-positive and corresponds to the self-improvement of language i. The second term captures cross-lingual interactions. If for any language $j \neq i$,

$$\langle \nabla_\theta \mathcal{L}_i(\theta), \nabla_\theta \mathcal{L}_j(\theta) \rangle > 0,$$

then updates driven by language j further decrease $\mathcal{L}_i$, resulting in positive transfer; negative inner products indicate interference.

This establishes that the rate of loss decrease for each language is governed by pairwise gradient alignment.

Algorithm 2 Compute Language Utilities and Costs from Gradients

1: Compute validation gradients $g_\ell(\theta)$ with respect to the last layer for all languages $\ell \in \{1, \dots, L\}$.
2: Compute pairwise cosine similarity matrix:

$$S_{\ell m}(\theta) = \frac{\langle g_\ell(\theta), g_m(\theta) \rangle}{\|g_\ell(\theta)\| \, \|g_m(\theta)\|}, \quad \forall \ell, m \in \{1, \dots, L\}.$$

3: Cluster languages using cosine distance $d_{\ell m}(\theta) = 1 - S_{\ell m}(\theta)$ to obtain language families $\{\mathcal{F}_k\}_{k=1}^K$.
4: *(After family selection , called with active family $\mathcal{F}_s$)*
5: Compute relative need of out-of-family languages:

$$\tilde{L}_j = \frac{L_j(\theta)}{\sum_{k \notin \mathcal{F}_s} L_k(\theta)}, \quad j \notin \mathcal{F}_s.$$

6: Compute gradient-based utility coefficients for $i \in \mathcal{F}_s$:

$$a_i = \max\left(0, \sum_{j \notin \mathcal{F}_s} S_{ij}(\theta) \cdot \|g_i(\theta)\| \cdot \|g_j(\theta)\| \cdot \tilde{L}_j\right),$$

then rescale: $a_i \leftarrow a_i / (\max_k a_k + \varepsilon) + 10^{-3}$.
7: Compute cost coefficients for $i \in \mathcal{F}_s$:

$$c_i' = \frac{1}{L_i(\theta) \cdot \|g_i(\theta)\|^2 + \varepsilon}, \qquad c_i = \frac{c_i'}{\sum_{k \in \mathcal{F}_s} c_k'}.$$

8: **return** language families $\{\mathcal{F}_k\}_{k=1}^K$ and, for active family, $\{a_i, c_i\}_{i \in \mathcal{F}_s}$.

Algorithm 3 Select Next Language Family via Combined Compatibility Score

Require: Language families $\{\mathcal{F}_k\}_{k=1}^K$, current parameters $\theta^{(r)}$, previous family mean gradient $\bar{g}_{\mathcal{F}_{\text{prev}}}$, weight α
1: Compute mean gradient for each family: $\bar{g}_{\mathcal{F}_k} = \frac{1}{|\mathcal{F}_k|} \sum_{\ell \in \mathcal{F}_k} g_\ell(\theta^{(r)})$.
2: **for** each valid family $\mathcal{F}_k$ (with remaining budget) **do**
3: Compute transition term: $T(\mathcal{F}_k) = \bar{g}_{\mathcal{F}_k} \cdot \bar{g}_{\mathcal{F}_{\text{prev}}} \; / \; (\|\bar{g}_{\mathcal{F}_{\text{prev}}}\| + \varepsilon)$.
4: Compute finite-difference term: $D(\mathcal{F}_k) = -\big[\mathcal{L}(\theta^{(r)} - \varepsilon \bar{g}_{\mathcal{F}_k}) - \mathcal{L}(\theta^{(r)})\big]/\varepsilon$.
5: **end for**
6: Min-max normalize T and D across all valid families to obtain $\tilde{T}$, $\tilde{D}$.
7: **for** each valid family $\mathcal{F}_k$ **do**
8: $C(\mathcal{F}_k \mid \theta^{(r)}) = \alpha \cdot \tilde{T}(\mathcal{F}_k) + (1 - \alpha) \cdot \tilde{D}(\mathcal{F}_k)$.
9: **end for**
10: Select next family: $\mathcal{F}_s^{(r)} = \arg\max_{\mathcal{F}_k} C(\mathcal{F}_k \mid \theta^{(r)})$.
11: Store $\bar{g}_{\mathcal{F}_s^{(r)}}$ as $\bar{g}_{\mathcal{F}_{\text{prev}}}$ for the next regime.
12: **return** $\mathcal{F}_s^{(r)}$

K Dynamic versus Static Data Allocation

Most prior work on multilingual data mixing computes fixed sampling ratios before training. Recent work (Li et al., 2025) theoretically motivates dynamic, batch-level data allocation. Here, we provide a complementary justification showing that *regime-dependent gradient geometry* necessitates dynamic, gradient-aware sampling.

Theorem K.1 (Regime-Dependent Gradient Geometry)**.** *Let $\theta \in \mathbb{R}^d$ be the parameters of a model trained on L languages, each with twice continuously differentiable loss $\{\mathcal{L}_\ell(\theta)\}_{\ell=1}^L$. Define per-language gradients*

$$g_\ell(\theta) := \nabla_\theta \mathcal{L}_\ell(\theta),$$

and the gradient matrix

$$G(\theta) := [\, g_1(\theta), g_2(\theta), \ldots, g_L(\theta) \,] \in \mathbb{R}^{d \times L}.$$

Suppose training proceeds in regimes. Let a regime updating a subset of languages $X \subseteq [L]$ modify parameters as

$$\theta' = \theta + \Delta\theta, \qquad \Delta\theta = -\eta \sum_{i \in X} \alpha_i g_i(\theta),$$

where $\eta > 0$ is the step size and α_i are regime-local sampling weights (not necessarily equal to the global weights w_i).

Then the post-regime gradients satisfy

$$g_i(\theta') = g_i(\theta) + \int_0^1 H_i(\theta + t\Delta\theta)\, \Delta\theta \, dt,$$

where $H_i(\theta) := \nabla^2_\theta \mathcal{L}_i(\theta)$ is the Hessian of language i.

Consequently,

$$\operatorname{span} G(\theta') \subseteq \operatorname{span}\Big(\{g_i(\theta)\}_{i=1}^L \;\cup\; \{H_i(\theta)\Delta\theta\}_{i=1}^L \Big).$$

If there exists any i such that

$$H_i(\theta)\Delta\theta \notin \operatorname{span}\{g_j(\theta)\}_{j=1}^L,$$

then

$$\operatorname{span} G(\theta') \neq \operatorname{span} G(\theta),$$

implying that gradient geometry changes across regimes.

L Continuous Knapsack with Diminishing Returns

We consider the following per-regime budget allocation problem over L languages:

$$\max_{\{x_i\}} \quad \sum_{i=1}^L a_i x_i^\beta \tag{18}$$

$$\text{s.t.} \quad \sum_{i=1}^L c_i x_i \le W, \tag{19}$$

$$\delta \le x_i \le U_i, \quad \forall i, \tag{20}$$

where x_i is the number of training samples allocated to language i, $a_i > 0$ is the utility of allocating samples to language i, $c_i > 0$ is the per-sample cost, $\delta \ge 0$ is the minimum allocation floor, $U_i \ge \delta$ is the per-language upper bound, $W \ge \delta \sum_i c_i$ is the total budget, and $0 < \beta < 1$ models diminishing returns. Since x^β is concave for $\beta \in (0, 1)$ and all constraints are linear, this is a convex optimization problem.

Lagrangian. Introduce Lagrange multipliers $\lambda \ge 0$ for the budget constraint, $\mu_i \ge 0$ for $x_i \ge \delta$, and $\nu_i \ge 0$ for $x_i \le U_i$. The Lagrangian is

$$\mathcal{L}(\mathbf{x}, \lambda, \mu, \nu) = \sum_{i=1}^L a_i x_i^\beta - \lambda\Big(\sum_{i=1}^L c_i x_i - W \Big) + \sum_{i=1}^L \mu_i (x_i - \delta) + \sum_{i=1}^L \nu_i (U_i - x_i). \tag{21}$$

Stationarity. Differentiating with respect to x_i gives

$$\frac{\partial \mathcal{L}}{\partial x_i} = a_i \beta x_i^{\beta - 1} - \lambda c_i + \mu_i - \nu_i = 0. \tag{22}$$

For interior solutions $\delta < x_i < U_i$, complementary slackness implies $\mu_i = \nu_i = 0$, yielding

$$a_i \beta x_i^{\beta - 1} = \lambda c_i. \tag{23}$$

Solving for x_i gives the unconstrained optimal allocation:

$$x_i = \left(\frac{a_i \beta}{\lambda c_i} \right)^{\frac{1}{1-\beta}}. \tag{24}$$

Optimal Solution. Accounting for both bounds, the optimal allocation is

$$x_i^* = \min\left\{U_i, \max\left\{\delta, \left(\frac{a_i\beta}{\lambda c_i}\right)^{\frac{1}{1-\beta}}\right\}\right\}. \tag{25}$$

If $\sum_{i=1}^{L} c_i U_i \le W$, the budget constraint is slack, $\lambda = 0$, and $x_i^* = U_i$. Otherwise, $\lambda > 0$ is chosen such that the budget is fully used:

$$\sum_{i=1}^{L} c_i x_i^* = W. \tag{26}$$

Since each x_i^* is non-increasing in λ, this equation can be solved efficiently via one-dimensional binary search.

Proposition L.1 (Pareto-Efficiency of Optimal Allocation)**.** *At the solution $\{x_\ell^*\}$ of Eq.* (9)*, the marginal utility per unit cost is equalized across all active languages:*

$$\frac{a_\ell\,\beta\,x_\ell^{\beta-1}}{c_\ell} = \lambda, \quad \forall\,\ell \in \mathcal{F}_s^{(r)} \text{ with } \delta < x_\ell^* < U_\ell, \tag{27}$$

while languages at the floor have marginal utility per unit cost at most λ and languages at their upper bound have it at least λ. Consequently, no marginal reallocation of budget from one language to another can improve the objective $\sum_\ell a_\ell x_\ell^\beta$, the allocation is Pareto-efficient with respect to the gradient-informed utility-cost tradeoff.

In the special case $c_\ell = 1$ and $U_\ell = \infty$ for all ℓ, with no language at the floor δ, the optimal allocation reduces to $x_\ell^* \propto a_\ell^{1/(1-\beta)}$, recovering a power-law scaling relationship without explicit curve fitting, subsuming the approaches of (Guo et al., 2025; Chen et al., 2023) as a special case.

M Licenses for Existing Assets

All assets used in this work are properly credited with citations to their original papers. We list the primary assets and their licenses below.

Datasets. Bactrian-X (Li et al., 2023) is released under CC BY-NC 4.0 (`https://huggingface.co/datasets/MBZUAI/Bactrian-X`). Aya (Singh et al., 2024) is released under Apache 2.0 (`https://huggingface.co/datasets/CohereForAI/aya_dataset`).

Models. Qwen3-4B (Yang et al., 2025) is released under Apache 2.0 (`https://huggingface.co/Qwen/Qwen3-4B-Base`). LLaMA-3.2-3B (Grattafiori et al., 2024) is released under the Llama 3.2 Community License (`https://huggingface.co/meta-llama/Llama-3.2-3B`). Gemma-3-4B (Gemma Team, 2025) is released under the Gemma Terms of Use (`https://huggingface.co/google/gemma-3-4b-pt`).

Evaluation Benchmarks. HellaSwag (Zellers et al., 2019) (MIT License); StoryCloze (Mostafazadeh et al., 2016) (no explicit license specified, `https://huggingface.co/datasets/Muennighoff/xstory_cloze`); MMLU-Pro (Wang et al., 2024b) (MIT License); ARC-E/ARC-C (Clark et al., 2018) (Apache 2.0); MultiNLU (Williams et al., 2018) (OANC License).

Baselines. GREATS (Wang et al., 2024a) is publicly available at `https://github.com/Jiachen-T-Wang/GREATS`; no explicit license file is present in the repository and it is used in accordance with the repository's terms. RegMix (Liu et al., 2025) is released under MIT License (`https://github.com/sail-sg/regmix`). PiKE (Li et al., 2025) was independently implemented following the paper's methodology, as no official repository is publicly available. MLP-MKC+ (Messmer et al., 2025) is used in accordance with its NeurIPS 2025 competition release terms.

All model and dataset versions correspond to those publicly available at the time of submission.

N Glossary of Notations Used

Symbol	Meaning
$\theta \in \mathbb{R}^d$	Shared model parameters.
$\theta^{(r)}$	Model parameters at the end of training regime r.
$\theta^{(t)}$	Model parameters after t optimization steps within a regime.
$L_\ell(\theta)$	Expected training loss for language ℓ.
$\mathcal{L}(\theta; x, y)$	Per-example loss function.
$L_{\mathcal{F}}(\theta)$	Validation loss for language group $\mathcal{F}$.
$\mathcal{V}_\ell$	Held-out validation set for language ℓ.
$\mathcal{V}_{\mathcal{F}}$	Union of validation sets for all languages in group $\mathcal{F}$.
$G_\ell(\theta)$	Validation gradient for language ℓ, defined as $\mathbb{E}_{(x,y)\sim\mathcal{V}_\ell}[\nabla_\theta \mathcal{L}(\theta; x, y)]$.
$S_{\ell m}(\theta)$	Cosine similarity between validation gradients of languages ℓ and m.
$S(\theta) \in \mathbb{R}^{L\times L}$	Gradient cosine-similarity matrix across all languages.
$d_{\ell m}(\theta)$	Cosine distance between gradients of languages ℓ and m.
$\Delta S^{(t)}$	Change in similarity matrix after t steps within a regime.
$\rho^{(t)}$	Relative drift statistic measuring change in gradient geometry within a regime.
τ_{rel}	Threshold for triggering a regime switch.
L	Total number of languages.
$\{\mathcal{F}_1, \ldots, \mathcal{F}_K\}$	Language families from gradient-based clustering.
$\mathcal{F}_k$	The k-th language group.
r	Index of training regimes.
$C(\mathcal{F} \mid \theta^{(r)})$	Combined compatibility score of group $\mathcal{F}$: convex combination of transition term $\tilde{T}$ and finite-difference term $\tilde{D}$.
$T(\mathcal{F}_k)$	Transition term: scalar projection of $\bar{g}_{\mathcal{F}_k}$ onto $\bar{g}_{\mathcal{F}_{\text{prev}}}$.
$D(\mathcal{F}_k)$	Finite-difference term: negated expected global loss change from a step in $\bar{g}_{\mathcal{F}_k}$ direction.
α	Weighting parameter for combined compatibility score (default 0.3).
$\bar{g}_{\mathcal{F}_k}$	Mean gradient of language family $\mathcal{F}_k$.
$\tilde{L}_j$	Relative validation loss of out-of-family language j: $L_j / \sum_{k\notin\mathcal{F}_s} L_k$.
δ	Minimum allocation floor per language in the knapsack: $\lfloor 0.05 \cdot W^{(r)}/\lvert\mathcal{F}_s\rvert \rfloor$.
x_i	Samples allocated to language i.
a_i	Utility coefficient: need-weighted gradient inner product with out-of-family languages, rescaled after clamping.
c_i	Cost coefficient: inverse of $L_i(\theta) \cdot \lVert g_i(\theta)\rVert^2$, normalized over active family $\mathcal{F}_s$.
U_i	Max allowable allocation.
W	Total training budget.
D_{rem}	Remaining budget.
β	Diminishing returns exponent.
λ	Lagrange multiplier.
η	Learning rate.